LeftOver

Review

- Web Service
  - Four requirements: data format, RPC mechanism, service description, discovery mechanism
  - SOAP WS: XML, SOAP, WSDL, UDDI
  - REST-Style: JSON or XML, REST-Style

JSON

- JavaScript Object Notation. Although it originates from JavaScript standard, it is a compute-rlaugnage neutral standard.
- A light-weight text-based data-interchange format.
  - Easy for humen to read and write.
  - Easy for machines to parse and generate.
- Corresponds to two data types in computer languages:
  - unordered list (name/value pairs) → object, struct, associative array, etc.
  - ordered list → array.
- It is very popular for data interchange between JavaScript on webpages and web services. JavaScript has a eval() function to directly translate JSON data into a native object, and therefore programmers do not need to worry how to parse JSON data format.
- For other languages, there are many JSON parsers that can be used.

http://json.org/

3

JSON

```
{
  "employees": [
    { "firstName":"John"  , "lastName":"Doe" },
    { "firstName":"Anna"  , "lastName":"Smith"},
    { "firstName":"Peter" , "lastName":"Jones"}
  ]
}
```

- In the example above, it describes an object (because of {…}). This object has one name/value pair. The name is "employees".
- The value paired with "employees" is an array (because of […]) containing three objects.
- Each of the contained object has two name/value pairs.
- Values can be nested structures (objects as an array element, array as the value for objects, etc.)

4

JSON

- Name is a string. JSON string mostly follow C/C++ string definitions, including some escape characters.
- Value can be string ("…"), object ({…}), array ([…]), number, true, false, null.

```
 1 {   "intArray": [1,2,3,4,5,6,7,8,9,10],
 2     "father": {
 3         "name": "George the Monkey", "age": 48,
 4         "married": false, "hasChildren": true,
 5         "speedTicket": null,
 6         "children": {
 7             "son1": { "name": "Lala", "age": 18 },
 8             "son2": { "name": "Kiki", "age": 15 },
 9             "daughter": { "name": "Koko", "age": 5 }
10         }
11     }
12 }
```

JSON

- JSON is different from XML in the following aspects:
  - There is no concept as "tag".
  - It is easier & faster to parse JSON than to parse XML.
  - JSON needs less data size to describe the same data.
- References
  - www.json.org
  - http://www.w3schools.com/json/json_intro.asp

6

WSDL
Web Service Description Language

- XML formatted.
- Describes SOAP web services.
- Content:
  - Description and format of messages that can be passed in <types> and <message> tags
  - Direction of message passingin<portType>:
    - Request-only, request-response, response-only
  - Message encoding in <binding> element (literal, etc.)
  - Location where service is offered in <service> element

7

Function Prototype

- Prototype in C/C++/Java:
  - Defines arguments/parameters and the returning information of functions
  - e.g.
    ```
    int func1(int a, int b, int c);
    ```
- Prototype in Web service
  - Web services provide functions through the Web
  - Prototype is required for each function provided through web services → WSDL!
  - In addition, the prototype also needs to identify where/how to invoke the function (e.g. URLs)

8

WSDL
Web Service Description Language

- WSDL is a document written in XML to describe a Web service. It:
  - Specifies the location of the service
  - Specifies the operations (or methods) the service exposes
  - Describes a set of SOAP messages and how the messages are exchanged
- Six main elements are there in a WSDL document:
  - <definition>: Root WSDL element
  - <types>: The data types used by the web service.
  - <message>: The messages used by the web service. → The input and output of a web service.
  - <portType>: The operations performed by the web service. Similar to defining classes and functions or methods in traditional programming.
  - <binding>: The communication protocols used by the web service
  - <service>: where the service is located

9

WSDL Example

```
<message name="getTermRequest">
   <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
   <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
      <input message="getTermRequest"/>
      <output message="getTermResponse"/>
  </operation>
</portType>
```

Similar to class

A method of class glossaryTerms

10

Online WSDLs & References

- Amazon
  - http://soap.amazon.com/schemas2/AmazonWebServices.wsdl
  - Details at: http://aws.amazon.com/
- eBay
  - http://developer.ebay.com/webservices/latest/eBaySvc.wsdl
- Microsoft TerraService
  - http://terraservice.net/TerraService2.asmx?WSDL
- References:
  - http://www.w3schools.com/wsdl/default.asp
  - http://www.w3.org/TR/wsdl
  - http://www.developer.com/services/article.php/1602051

11

SOAP
Simple Object Access Protocol

- A simple XML based protocol to let applications exchange information over HTTP → web services, communication between applications, platform independent.
- XML over HTTP
  - HTTP – for data transport, some headers are added to HTTP protocol
  - XML as data transmission format
- Key element of SOAP:
  - New MIME type: text/xml
  - Agreed definitions of data types, mandatory values, etc.
- Use URL to describe endpoints (how to access services)

12

SOAP Building Blocks

- A SOAP message is an ordinary XML document containing the following elements:
  - A required Envelope element that identifies the XML document as a SOAP message
  - An optional Header element that contains header information
  - A required Body element that contains call and response information
  - An optional Fault element that provides information about errors that occurred while processing the message
- The message is delivered to web server (service provider) through HTTP POST method
- The response from web services is gotten from server using HTTP GET method.

13

SOAP Example
(Sending)

```
POST /InStock HTTP/1.1
Host: www.stock.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

  <soap:Body xmlns:m="http://www.stock.org/stock">
    <m:GetStockPrice>
      <m:StockName> IBM </m:StockName>
    </m:GetStockPrice>
  </soap:Body>

</soap:Envelope>
```

14

SOAP Example
(Response)

```
HTTP/1.1 200 OK
Content-Type: application/soap; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

  <soap:Body xmlns:m="http://www.stock.org/stock">
    <m:GetStockPriceResponse>
      <m:Price> 34.5 </m:Price>
    </m:GetStockPriceResponse>
  </soap:Body>

</soap:Envelope>
```

15

Reference

- http://www.w3.org/TR/soap12-part0/
- http://www.w3schools.com/soap/default.asp
- http://terraservice.net/
  - http://terraservice.net/TerraService2.asmx
  - http://terraservice.net/TerraService2.asmx?WSDL
- http://livedocs.adobe.com/jrun/4/Programmers_Guide/wsmonitor3.htm
- http://www.intertwingly.net/stories/2002/03/16/aGentleIntroductionToSoap.html
- http://www.soapware.org/bdg

16

UDDI
Universal Description, Discovery, and Integration

- Defines a standard of being yellow pages (directory service) for web service providers.
- UBR: UDDI Business Registry
  - Global public directory of businesses and services

17

Some Online Web Services

- http://www.google.com/apis/
- http://aws.amazon.com/
- http://msdn.microsoft.com/en-us/library/dd877180.aspx
- http://arcweb.esri.com/arcwebonline/index.htm
- http://developer.ebay.com/

18

Cloud Computing

- Web services started as a tool to integrate heterogeneous information systems.
- Now it becomes a way to access "Cloud Resources" …

19

Lecture 16

Security
Review

Internet Security

Security

- We've developed two applications
  - Two-tier (C# ←→ database)
  - Three-tier (Browser ←→ web server ←→ database)
- Two-tier applications usually reside in the intranet (corporate network isolated from the Internet), thus security of such application is less of a problem (unless?)
- For the three-tier architecture, there is multi-point of vulnerability.
  - Browser (Client)
  - Network (Connection between clients and servers)
  - Web server (Server)

22

Three Premises for Web Security

- User (browser) premises
  - Remote server is operated by organization stated
  - Documents returned by server are free from viruses, etc.
  - Remote server will not distribute user's private info, such as Web use
- Webmaster (server) premises
  - User will not attempt to break into or alter contents of Web site
  - User will not try to gain access to documents that he/she is not allowed
  - User will not try to crash the server or deny service to others
  - If user has identified him/herself, user is who he/she claims to be
- Network premises (user and Webmaster)
  - Network is free from 3rd party eavesdroppers
  - Network delivers information intact, not tampered with by 3rd parties

Reference: MIT 1.264 lecture notes

23

Client Risks

- Content: applets, scripts, ActiveX controls, plug-ins, images
  - Browsers download and run software without advance notice
    - User usually cannot virus-check before using
  - Formerly innocuous content such as Word files can start viruses
- Privacy loss
  - Web server site logs: IP address, document retrieved, date/time, previous URL, and more.
  - Cookies. Have been abused by marketing to track user habits
    - http://www.cookiecentral.com/dsm.htm
  - Surreptitious email with private info
    - Microsoft Outlook has been repeatedly attacked via email.

Reference: MIT 1.264 lecture notes

24

Server Risks

- Webjacking
  - Break-in and modification of site
  - Hundreds reported, including NASA, CIA, USAF, …
  - Exploit operating system and email holes, poor configuration, …
- Denial of service
  - Attacks that cause system to expend large resources in response

Reference: MIT 1.264 lecture notes

25

Network Risks

- Packet sniffers: look for passwords, credit card numbers
  - Kits available on Internet
  - Small programs, installed on compromised computer in network
  - Cryptography is technical solution to sniffing
- IP address spoofing: pretend you're another machine
  - Look-alike sites set up with stolen pages, etc.
    - Mimic Web merchant, bank, etc.
  - Authentication (digital certificates) are technical solution

Reference: MIT 1.264 lecture notes

26

Cryptography and Encryption

- Plaintext: original message
- Ciphertext: encrypted message
- Algorithm: function converting plaintext to ciphertext
- Key: number used by algorithm to encrypt and/or decrypt
- Example
  - Plaintext: 11223344
  - Key: 44332211
  - Algorithm: (symmetric)
    - Encryption: Ciphertext = Plaintext + Key
    - Decryption: Plaintext = Ciphertext - Key
  - Ciphertext: 55555555

Reference: MIT 1.264 lecture notes

27

Symmetric Algorithms (1/2)

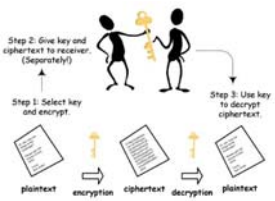
- Symmetric algorithms
  - Use the same key for encryption and decryption (as in the previous example)
  - Several algorithms
    - DES (Data Encryption Standard): 56 bit key, in common use
      - Splits data into pieces, XORs, reshuffles
      - Cracked in two days in June, 1998
    - Triple DES: encrypt/decrypt/encrypt with 2 or 3 DES keys: 168 bit effective key length
      - Encrypt with key 1, decrypt with key 2, encrypt with key 1 again
    - Rijndael: Current US government AES standard
    - RC2, RC4, RC5: 40 (export)-2048 bit keys, in common use by encrypting Web servers and browsers
    - IDEA: 128 bit key, popular in Europe, used in PGP, patented
    - Blowfish, Twofish: up to 448 bit key, unpatented, use increasing

Reference: MIT 1.264 lecture notes

28

Symmetric Algorithms (2/2)

- Problems with symmetric keys
  - Must be exchanged in advance, via secure method
  - Multi-way communication infeasible:
    - If many users must communicate with server, compromising any one can compromise all (since they all use the same key)



Reference: MIT 1.264 lecture notes

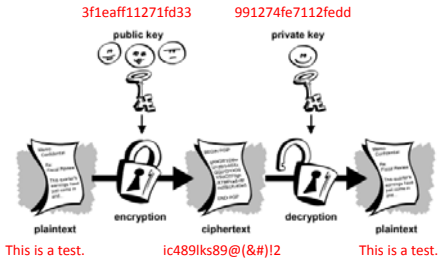
29

Asymmetric or "public key" algorithms (1/2)

- Two keys are used in these algorithms: public key and private/secret key
  - The public key is used for encryption
  - The secret key is used for decryption
  - These two keys are obviously related, but a good algorithm makes sure
    - Only the corresponding private key can be used to decrypt the message
    - Only the corresponding public key can be used to encrypt the message
    - It is impossible to deduce the private key from the public key

30

Asymmetric Algorithm



Asymmetric or "public key" algorithms (2/2)

- Some algorithms
  - RSA: 512-1024 bit, in common use for Web and email
  - ElGamal: unpatented, infringement on Diffie-Hellman (inventors of public key algorithms) expired 1997
- Problems with public key algorithms
  - Speed: RSA is 1000 times slower than symmetric algorithms
  - Problem avoided by using RSA to exchange a symmetric "session key" and then using symmetric method for the rest of the session.

Reference: MIT 1.264 lecture notes

32

Key Length

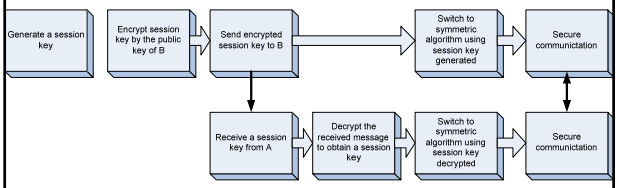
- Assume:
  - Algorithm is good
  - Algorithm coded correctly
  - Key management is correct and secure
- Then only way to crack message is brute force
  - Public key lengths must be longer than symmetric to provide same level of security (why?)
    - 384 bit public key offers same security as 40 bit symmetric key (not much)
    - Public keys should be at least 1024

Symmetric key length	Time to crack on PCs	Time to crack on big iron
40 bits	Seconds	Milliseconds
56 bits	Days	Hours
64 bits	Months	Days
80 bits	Million of Years	Thousands of Years

Reference: MIT 1.264 lecture notes

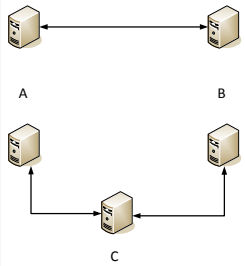
33

To use asymmetric algorithm efficiently



IF the public key of B is authentic, then the communication is secure.

Man-in-the-middle Attack (1/3)



A:
1. Hello, I need to login
2. Bob
3. myPassword
. . .

B:
1. What is your username
2. Bob, what is your password
3. What do you want to do.
. . .

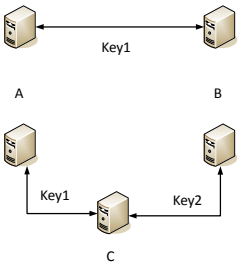
C:
1a. [Get message from A, forward to B]
1b. [Get response from B, forward to A]
2a. [Get message from A, forward to B]
2b. [Get response from B, forward to A]
. . .

A:
1. Hello, I need to login
2. Bob
3. myPassword
. . .

B:
1. What is your username
2. Bob, what is your password
3. What do you want to do.
. . .

35

Man-in-the-middle Attack (2/3)



A:
1. Exchange key1
2. {Hello, I need to login.}
3. {Bob}
4. {myPassword}
. . .

B:
1. Exchange key1
2. {What is your username}
3. {Bob, what is your password}
4. {What do you want to do.}
. . .

B:
1. Exchange key2 with C
2. {What is your username}
3. {Bob, what is your password}
4. {What do you want to do.}
. . .

C:
1a. Exchange key1 with A, Exchange key2 with B
1b. {[Get response from B, forward to A]}
2a. {[Get message from A, forward to B]}
2b. {[Get response from B, forward to A]}
. . .

A:
1. Exchange key1
2. {Hello, I need to login}
3. {Bob}
4. {myPassword}
. . .

{...} means encrypted message

36

Man-in-the-middle Attack (3/3)

- The above attack only works
  - IF A doesn't know B's true public key in advance,
  - AND cannot check if the public key it gets from network is indeed.
  - If A knows B's public key in advance, or A can check the public key it gets from the network does indeed come from B, then the attack fails.
- To check if a document (public key) if authentic → Digital Signature

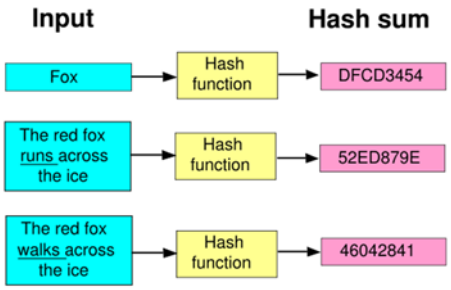
Digital Signatures (1/2)

- The key pairs (public vs. secret) are interchangeable
  - Plaintext → public key → ciphertext → secret key → plaintext
  - Plaintext → secret key → ciphertext → public key → plaintext
- To digitally sign a document
  1. Use a hashing function (mathematical function) to digest/summary the document into a few bits (message digest)
  2. Encrypt the message digest with one's private key (e.g. Bob)
  3. Append the original document with the encrypted message digest

Good reference: http://www.youdzone.com/signature.html

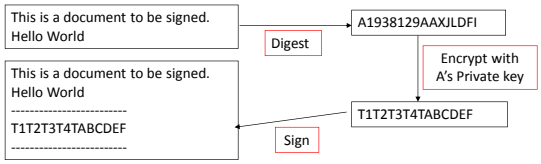
38

Hashing



http://en.wikipedia.org/wiki/Hash_function

Example (Signing)



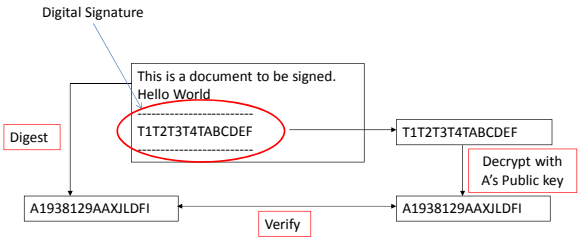
40

Digital Signatures (2/2)

- To determine if one has signed the document
  1. Find the digital signature and decrypt it with Bob's public key → signed message digest: X
  2. Calculate the message digest using the same hashing function → calculated message digest Y
  3. If X matches Y, then the document has been signed by Bob
- How do we know that we have the correct Bob's public key?

41

Example (Verification)



42

Public Key Infrastructure, PKI

- A system that enables users of a public network to exchange data securely and privately through the use of a public and private cryptographic key pair that is obtained and shared through a trusted authority.
  - Digital certificate
  - Certificate Authority

43

Digital Certificate (1/2)

- Digital certificates are issued by CA (Certificate Authority)
  - Contains information about the certificate holder, including his/her public key, issuer's information, validity, figure-prints of the certificate, etc.
  - CA's authentic public key known to all users of digital certificates
  - Thus, the validity of the public key on a digitally signed document can be verified using public keys of digital certificates authorized by CA

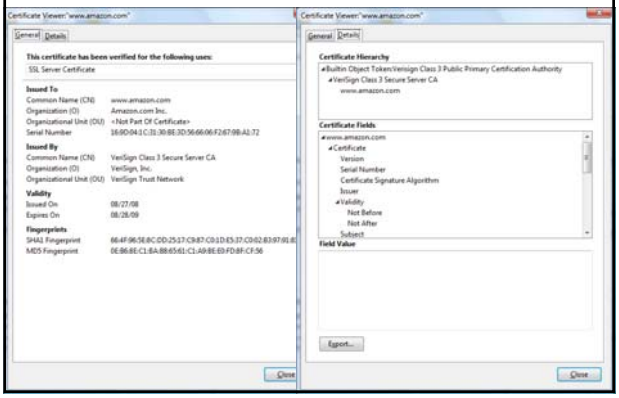
44

Digital Certificate (2/2)

- Thus, digitally signed documents by Bob can be verified
  - Check if I have Bob's digital certificate. If not, ask Bob send his digital certificate to me.
  - Use CA's public key to decrypt Bob's digital certificate to obtain Bob's public key.
  - Find the digital signature on the document and decrypt it with Bob's public key → signed message digest: X
  - Calculate the message digest using the same hashing function calculated message digest Y
  - If X matches Y, then the document has been signed by Bob

45

Digital Certificate of Amazon.com



Review

- There are multiple points of attacks in a three-tier information system architecture
  - Web server
  - Network
  - Web client
- A capable computer administrator can take care security updates on OS, Web server software, web browser, etc. …
  - He/she has no control on the network, especially the Internet.
  - Man in the middle attack!

Review

- To defend/defeat man-in-the-middle
  - Symmetric encryption algorithms alone is not feasible
  - Asymmetric encryption algorithms are too slow
  - Generate a session symmetric key, transfer it securely using asymmetric algorithms, and then switch to symmetric encryption
  - Problem: how to get authenticate public key ?
  - Solution: digital signature, digital certificate, and PKI
- With PKI, Internet can be secured. As a result, Internet commerce (e-commerce) blooms.